

文件上传

[导入jar包，或者maven](#)



[使用类的介绍](#)

[FileItem类](#)

[ServletFileUpload类](#)

[代码编写](#)

导入jar包，或者maven

1.  `commons-fileupload-1.5.jar`
 `commons-io-2.11.0.jar`

```
XML |
1  <!-- https://mvnrepository.com/artifact/commons-fileupload/commons-fileupload -->
2  <dependency>
3      <groupId>commons-fileupload</groupId>
4      <artifactId>commons-fileupload</artifactId>
5      <version>1.5</version>
6  </dependency>
7  <!-- https://mvnrepository.com/artifact/commons-io/commons-io -->
8  <dependency>
9      <groupId>commons-io</groupId>
10     <artifactId>commons-io</artifactId>
11     <version>2.11.0</version>
12 </dependency>
13
```

使用类的介绍

【文件上传的注意事项】

1. 为保证服务器安全，上传文件应该放在外界无法直接访问的目录下，比如放于web-inf目录下。
2. 为防止文件覆盖的现象发生，要为上传文件产生一个唯一的文件名
3. 要限制上传文件的最大值。
4. 可以限制上传文件的类型，在收到上传文件名时，判断后缀名是否合法。

【需要用到的类详解】

ServletFileUpload负责处理上传的文件数据,并将表单中每个输入项封装成一个filetem对象, 在使用ServletFileUpload对象解析请求时需要DiskFileItemFactory对象。所以, 我们需要在进行解析工作前构造好DikFileItemFactory对象, 通过ServletFileUpload对象的构造方法或setFileItemFactory() 设置ServletFileUpload对象的fileItemFactory属性。

FileItem类

在HTML页面input必须有 `name<input type="file" name="filename">`

表单如果包含一个文件上传输入项的话, 这个表单的enctype属性就必须设置为 `multipart / form-data`

浏览器表单的类型如果为 `multipart / form-data`, 在服务器想获取数据就要通过流

【常用方法介绍】

```
1  //isFormField 方法用于判断FileItem类对象封装的数据是一个普通文本表单
2  //还是一个文件表单, 如果是普通表单字段则返回true, 否则返回false
3  boolean isFormField();
4
5  //getFieldName方法用于返回表单标签name属性的值
6  String getFieldName();
7  //getString 方法用于将FileItem对象中保存的数据流内容以一个字符串返回
8  String getString();
9
10 //getName 方法用于获得文件上传字段中得文件名
11 String getName();
12
13 //以流得形式返回上传文件得数据内容
14 InputStream getInputStream();
15
16 //delete方法用来清空FileItem类对象中存放的主体内容
17 //如果主题内容被保存在临时文件中, delete方法将删除临时文件
18 void delete();
```

ServletFileUpload类

ServletFileUpload负责处理上传的文件数据，并将表单中每个输入项封装成一个FileItem对象中，使用其paraRequest(HttpServletRequest)方法可以将通过表单中每一个HTML标签提交的数据封装成一个FileItem对象，然后以List列表的形式返回，使用该方法处理上传文件简单易用。

代码编写

UploadFileServlet

```
package com.yatian.servlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.util.List;
import java.util.UUID;
import org.apache.commons.fileupload.FileItem;
import org.apache.commons.fileupload.FileUploadException;
import org.apache.commons.fileupload.ProgressListener;
import org.apache.commons.fileupload.disk.DiskFileItemFactory;
import org.apache.commons.fileupload.servlet.ServletFileUpload;

public class FileServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request,
        HttpServletResponse response) throws javax.servlet.ServletException, IOException {
        //判断上传的表单是普通表单还是带文件的表单，是返回true,否返回false;

```

```
        if (!ServletFileUpload.isMultipartContent(request)){return; //如果这是一个普通文件我们直接返回

```

```
        } //如果通过了这个if，说明我们的表单是带文件上传的 //创建上传文件的保存目录，为了安全建议在WEB-INF目录下，用户无法访问

```

```
        String uploadpath = this.getServletContext().getRealPath("WEB-INF/Upload"); //获取上传文件的保存路径

```

```
        File uploadfile = new File(uploadpath);
        if (!uploadfile.exists()){

```

```
            uploadfile.mkdir(); //如果目录不存在就创建这样一个目录

```

```
        } //临时文件

```

//临时路径，如果上传的文件超过预期的大小，我们将它存放到一个临时目录中，过几天自动删除，或者提醒用户转存为永久

```
String tmppath = this.getServletContext().getRealPath("WEB-INF/tmp");
```

```
File file = new File(tmppath);
```

```
if (!file.exists()){
```

```
file.mkdir();//如果目录不存在就创建这样临时目录
```

```
}*/
```

//处理上传的文件一般需要通过流来获取，我们可以通过request.getInputStream(),原生态文件上传流获取，十分麻烦//但是我们都建议使用Apache的文件上传组件来实现，common-fileupload,它需要依赖于common-io组件；

```
try{//1、创建DiskFileItemFactory对象，处理文件上传路径或限制文件大小
```

```
DiskFileItemFactory factory =getDiskFileItemFactory(file);//2、获取ServletFileUpload
```

```
ServletFileUpload upload =getServletFileUpload(factory);//3、处理上传文件
```

```
String msg =uploadParseRequest(upload,request,uploadpath);//Servlet请求转发消息
```

```
request.setAttribute("msg",msg);
```

```
request.getRequestDispatcher("/info.jsp").forward(request,response);
```

```
}catch(FileUploadException e){  
    e.printStackTrace();  
  
}
```

```
}public staticDiskFileItemFactory gteDiskFileItemFactory(File file){//1、创建DiskFileItemFactory对象，处理文件上传路径或限制文件大小
```

```
DiskFileItemFactory factory = newDiskFileItemFactory();//通过这个工厂设置一个缓冲区，当上传的文件大小大于缓冲区的时候，将它放到临时文件中；
```

```
factory.setSizeThreshold(1024 * 1024);//缓冲区大小为1M
```

```
factory.setRepository(file);returnfactory;
```

```
}public staticServletFileUpload getServletFileUpload(DiskFileItemFactory factory){//2、获取ServletFileUpload
```

```
ServletFileUpload upload = newServletFileUpload(factory);*//监听文件上传进度
```

```
upload.setProgressListener(new ProgressListener() {
```

```
@Override
```

```
public void update(long pBytesRead, long lpContentLenght, int i) {
```

```
//pBytesRead:已读取到的文件大小
```

```
//pContentLenght: 文件大小
```

```
System.out.println("总大小: "+lpContentLenght+"已上传: "+pBytesRead);
```

```

}

});

//处理乱码问题

upload.setHeaderEncoding("UTF-8");

//设置单个文件的最大值

upload.setFileSizeMax(1024 * 1024 * 10);

//设置总共能够上传文件的大小

//1024 = 1kb * 1024 = 1M * 10 = 10M

upload.setSizeMax(1024 * 1024 * 10);*/

return upload;

}public static String uploadParseRequest(ServletFileUpload upload,HttpServletRequest
request,String uploadpath) throwsIOException, FileUploadException { String msg= "";//3、处理上
传文件//把前端的请求解析，封装成一个FileItem对象 List fileItems
=upload.parseRequest(request);for(FileItem fileItem : fileItems) {if (fileItem.isFormField()){ //判断是
普通表单还是带文件的表单//getFieldName指的是前端表单控件的名字 String name
=fileItem.getFieldName(); String value= fileItem.getString("UTF-8");//处理乱码
System.out.println(name+":"+value); }else {//判断它是带文件的表单//=====
处理文件=====// //拿到文件的名称 String uploadFileName
=fileItem.getName(); System.out.println("上传的文件名： "+uploadFileName);if

```

```

(uploadFileName.trim().equals("") || uploadFileName == null){continue; }//获得上传的文件名, 例如/img/girl/ooa.jpg,只需要ooa, 其前面的后面的都不需要 String fileName =
uploadFileName.substring(uploadFileName.lastIndexOf("/") + 1);//获得文件的后缀名 String
fileExtName = uploadFileName.substring(uploadFileName.lastIndexOf(".") + 1);/** 如果后缀名
fileExtName 不是我们需要的 *就直接return, 不处理, 告诉用户类型不对 **/System.out.println("文件
信息 【文件名: "+fileName+"文件类型: "+fileExtName+"】 ");//可以使用UUID(唯一通用识别码)来保
证文件名的统一 String uuidFileName
=UUID.randomUUID().toString();//=====传输文件
=====// //获得文件上传的流 InputStream inputStream
=fileItem.getInputStream();//创建一个文件输出流 FileOutputStream fos = new
FileOutputStream(uploadpath + "/" + uuidFileName + "." + fileExtName);//创建一个缓冲区 byte[]
buffer = new byte[1024 * 1024];//判断是否读取完毕 int len = 0;//如果大于0, 说明还存在数据 while
((len=inputStream.read(buffer))>0){ fos.write(buffer,0,len); }//关闭流 fos.close();
inputStream.close(); msg= "文件上传成功! "; fileItem.delete();//上传成功, 清除临时文件 }
}returnmsg; } }

```

```
1  //package com.cong.servlet;
2  //
3  //import org.apache.commons.fileupload.FileItem;
4  //import org.apache.commons.fileupload.FileUploadException;
5  //import org.apache.commons.fileupload.disk.DiskFileItemFactory;
6  //import org.apache.commons.fileupload.servlet.ServletFileUpload;
7  //
8  //import javax.servlet.ServletException;
9  //import javax.servlet.http.HttpServletRequest;
10 //import javax.servlet.http.HttpServletResponse;
11 //import java.io.File;
12 //import java.io.FileOutputStream;
13 //import java.io.IOException;
14 //import java.io.InputStream;
15 //import java.util.List;
16 //import java.util.UUID;
17 //
18 //public class FileServlet extends javax.servlet.http.HttpServlet{
19 //    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
20 //
21 //        //判断上传的文件是普通表单还是带文件的表单
22 //        if(!ServletFileUpload.isMultipartContent(request)){
23 //            return;//终止方法运行，说明这是一个普通的表单，直接返回
24 //        }
25 //
26 //        //创建上传文件的保存路径，建议在WEB-INF路径下，安全，用户无法直接访问上传的文件；
27 //        String uploadPath = this.getServletContext().getRealPath("/WEB-INF/upload");
28 //        File uploadFile = new File(uploadPath);
29 //        if(!uploadFile.exists()){
30 //            uploadFile.mkdir();//创建这个目录
31 //        }
32 //
33 //        //缓存，临时文件
34 //        //临时路径，假如文件超过了预期的大小，我们就把他放到一个临时文件中，过几天自动删除，或者提醒用户转存为永久
35 //        String tmpPath = this.getServletContext().getRealPath("/WEB-INF/tmp");
36 //        File file = new File(tmpPath);
37 //        if(!file.exists()){
38 //            file.mkdir();//创建这个临时文件目录
39 //        }
40 //
```



```

41 //          //处理上传的文件，一般都需要通过流来获取，我们可以使用request.getInputS
tream() ,原生态的文件上传流获取，十分麻烦
42 //          //但是我们都建议使用Apache的文件上传组件来实现，common-fileupload，它
需要依赖于commons-io组件
43 //
44 //          //1.创建DiskFileItemFactory对象，处理文件上传路径或者大小的限制的：
45 //          DiskFileItemFactory factory = new DiskFileItemFactory();
46 //          /*
47 //          //通过这个工厂设置一个缓冲区，当上传的我呢见大于这个缓冲区的时候，将他放到
临时文件中；
48 //          factory.setSizeThreshold(1024*1024); //缓冲区大小为1M
49 //          factory.setRepository(file); //临时目录的保存目录，需要一个File
50 //          */
51 //
52 //          //2.获取ServletFileUpload
53 //          ServletFileUpload upload = new ServletFileUpload(factory);
54 //
55 //          //3.处理上传的文件
56 //          try {
57 //              String msg = uploadParseRequest(upload,request,uploadPath);
58 //              //servlet请求转发消息
59 //              request.setAttribute("msg", msg);
60 //              request.getRequestDispatcher("/info.jsp").forward(request,r
esponse);
61 //          }catch (FileUploadException e){
62 //              e.printStackTrace();
63 //          }
64 //
65 //
66 //
67 //
68 //          //3.处理上传的文件
69 //          //把前端请求解析，封装成一个FileItem对象，需要从ServletFileUplo
ad对象中获取
70 //          try {
71 //              List<FileItem> fileItems = upload.parseRequest(request);
72 //              //fileItem每一个表单对象
73 //              for (FileItem fileItem : fileItems) {
74 //                  //判断上传的文件是普通的表单还是带文件的表单
75 //                  if (fileItem.isFormField()) {
76 //                      //getFileName指的是前端表单控件的名字；
77 //                      String name = fileItem.getFieldName();
78 //                      String value = fileItem.getString("UTF-8");//处理
乱码
79 //                      System.out.println(name + ":" + value);
80 //                  }else{//文件
81 //                      //=====处理文件=====//
82 //                      //拿到文件的名字
83 //                      String uploadFileName = fileItem.getName();

```

```

83      ////          //可能出现文件名不合法的情况
84      ////          if(uploadFileName.trim().equals("") || uploadFile
Name==null){
85      ////          continue;
86      ////          }
87      ////          //获得上传的文件名 /images/girl/paojie.png
88      ////          String fileName = uploadFileName.substring(upload
FileName.lastIndexOf("/")+1);
89      ////          //获得文件的后缀名
90      ////          String fileExtName = uploadFileName.substring(upload
FileName.lastIndexOf(".")+1);
91      ////
92      ////          //网络传输中的东西，都需要序列化
93      ////          //POJO，实体类，如果想要早多个电视上运行，          传输===
=> 需要把对象都序列化了
94      ////          //implements Serializable : 标记接口          jvm---->
本地方法栈          native----> C++
95      ////
96      ////
97      ////
98      ////          //可以使用UUID（唯一识别的通用码），保证文件名唯一；
99      ////          //UUID.randomUUID()，随机生一个唯一识别的通用码
100      ////          String uuidPath = UUID.randomUUID().toString();
101      ////
102      ////          //=====存放地址=====//
103      ////          //存到哪? uploadPath
104      ////          //文件真实存在的路径realPath
105      ////          String realPath = uploadPath+"/"+uuidPath;
106      ////          //给每个文件创建一个对应的文件夹
107      ////          File realPathFile = new File(realPath);
108      ////          if(!realPathFile.exists()){
109      ////              realPathFile.mkdir();
110      ////          }
111      ////
112      ////          //=====文件传输=====//
113      ////          //获得文件上传的流
114      ////          InputStream inputStream = fileItem.getInputStream
();
115      ////
116      ////          //创建一个文件输出流
117      ////          //realPath = 真实的文件夹
118      ////          //差了一个文件；加上输出文件的名字+"/"+uuidFileName
119      ////          FileOutputStream fos = new FileOutputStream(realP
ath+"/"+fileName);
120      ////
121      ////          //创建一个缓冲区
122      ////          byte[] buffer = new byte[1024*1024];
123      ////
124      ////          //判断是否读取完毕

```

```

124     int len = 0;
125     //如果大于0说明还存在数据;
126     while((len = inputStream.read(buffer))>0){
127         fos.write(buffer, 0, len);
128     }
129
130     //关闭流
131     fos.close();
132     inputStream.close();
133
134     msg = "文件上传成功";
135     fileItem.delete();//上传成功, 清除临时文件
136 }
137 return msg;
138 }
139 //
140 //
141 // }
142 //
143 // protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
144 //
145 // }
146 // public static DiskFileItemFactory gteDiskFileItemFactory(File file)
147 // {
148 //     //1、创建DiskFileItemFactory对象, 处理文件上传路径或限制文件大小
149 //     DiskFileItemFactory factory = new DiskFileItemFactory();
150 //     //
151 //     //通过这个工厂设置一个缓冲区, 当上传的文件大小大于缓冲区的时候, 将它放到临时文件中;
152 //     factory.setSizeThreshold(1024 * 1024);//缓冲区大小为1M
153 //     //
154 //     factory.setRepository(file);
155 //     //
156 //     return factory;
157 // }
158 // }
159 //
160 // public static String getServletFileUpload(DiskFileItemFactory factory){
161 //     //2、获取ServletFileUpload
162 //     ServletFileUpload upload = new ServletFileUpload(factory);
163 //     //
164 //     //监听文件上传进度
165 //     upload.setProgressListener(new ProgressListener() {
166
167
168

```

```

169 // @Override
170 //
171 // public void update(long pBytesRead, long lpContentLenght, int i) {
172 // /// pBytesRead: 已读取到的文件大小
173 //
174 // /// lpContentLenght: 文件大小
175 //
176 // System.out.println("总大小: "+lpContentLenght+"已上传: "+pBytesRead);
177 //
178 // }
179 //
180 // });
181 //
182 // /// 处理乱码问题
183 //
184 // upload.setHeaderEncoding("UTF-8");
185 //
186 // /// 设置单个文件的最大值
187 //
188 // upload.setFileSizeMax(1024 * 1024 * 10);
189 //
190 // /// 设置总共能够上传文件的大小
191 //
192 // /// 1024 = 1kb * 1024 = 1M * 10 = 10M
193 //
194 // upload.setSizeMax(1024 * 1024 * 10);
195 //
196 // return upload;
197 //
198 // }*/
199 //
200 // public static String uploadParseRequest(ServletFileUpload upload,
201 //     HttpServletRequest request, String uploadpath) throws IOException, FileU
202 //     ploadException {
203 //     String msg = "";
204 //
205 //     /// 3、处理上传文件
206 //
207 //     /// 把前端的请求解析，封装成一个FileItem对象
208 //
209 //     List fileItems = upload.parseRequest(request);
210 //
211 //     for (FileItem fileItem : fileItems) {
212 //         if (fileItem.isFormField()) { // 判断是普通表单还是带文件的表
213 //             单
214 //
215 //             /// getFieldname指的是前端表单控件的name
216 //

```

```

214 //          String name = fileItem.getFieldName();
215 //
216 //          String value = fileItem.getString("UTF-8");//处理乱
217 码
218 //
219 //          System.out.println(name+":"+value);
220 //
221 //          }else { //判断它是带文件的表单
222 //
223 //          ////=====处理文件=====//
224 //
225 //          ////拿到文件的名字
226 //
227 //          String uploadFileName = fileItem.getName();
228 //
229 //          System.out.println("上传的文件名: "+uploadFileName);
230 //
231 //          if (uploadFileName.trim().equals("") || uploadFileN
232 ame == null){
233 //              continue;
234 //          }
235 //
236 //          ////获得上传的文件名, 例如/img/girl/ooa.jpg, 只需要ooa, 其前面的后面的都不需要
237 //
238 //          String fileName = uploadFileName.substring(uploadFi
239 leName.lastIndexOf("/") + 1);
240 //
241 //          ////获得文件的后缀名
242 //
243 //          String fileExtName = uploadFileName.substring(uploa
244 dFileName.lastIndexOf(".") + 1);
245 //
246 //          /*
247 //
248 //          * 如果后缀名 fileExtName 不是我们需要的
249 //
250 //          *就直接return, 不处理, 告诉用户类型不对
251 //
252 //          * */
253 //
254 //          System.out.println("文件信息 【文件名: "+fileName+"文件
255 类型: "+fileExtName+"】 ");
256 //
257 //          ////可以使用UUID(唯一通用识别码)来保证文件名的统一
258 //
259 //          String uuidFileName = UUID.randomUUID().toString();
260 //

```

```

257  /////=====传输文件=====//
258  //
259  /////获得文件上传的流
260  //
261  //          InputStream inputStream = fileItem.getInputStream
262  ();
263  //
264  /////创建一个文件输出流
265  //
266  //          FileOutputStream fos = new FileOutputStream(uploadp
267  ath + "/" + uuidFileName + "." + fileExtName);
268  //
269  /////创建一个缓冲区
270  //
271  //          byte[] buffer = new byte[1024 * 1024];
272  //
273  /////判断是否读取完毕
274  //
275  //          int len = 0;
276  //
277  //          while ((len=inputStream.read(buffer))>0){
278  //              fos.write(buffer,0,len);
279  //          }
280  //
281  //
282  //          }
283  //
284  //          fos.close();
285  //
286  //          inputStream.close();
287  //
288  //          msg = "文件上传成功! ";
289  //
290  //          fileItem.delete();//上传成功，清除临时文件
291  //
292  //          }
293  //
294  //          }
295  //
296  //          return msg;
297  //
298  //      }
299  //
300  //  }
301  //}
302  package com.cong.servlet;

```

```

303
304 import org.apache.commons.fileupload.FileItem;
305 import org.apache.commons.fileupload.FileUploadException;
306 import org.apache.commons.fileupload.disk.DiskFileItemFactory;
307 import org.apache.commons.fileupload.servlet.ServletFileUpload;
308
309 import javax.servlet.ServletException;
310 import javax.servlet.annotation.WebServlet;
311 import javax.servlet.http.*;
312 import java.io.*;
313 import java.nio.file.Paths;
314 import java.util.List;
315 import java.util.UUID;
316
317 @WebServlet("/upload") // 定义 Servlet 映射路径
318 public class FileServlet extends HttpServlet {
319
320     private static final long serialVersionUID = 1L;
321
322     // 设置最大文件大小为 10MB
323     private static final int MAX_FILE_SIZE = 10 * 1024 * 1024;
324
325     // 设置内存缓冲区大小为 1MB
326     private static final int MEMORY_THRESHOLD = 1 * 1024 * 1024;
327
328     // 上传文件存储目录
329     private static final String UPLOAD_DIRECTORY = "WEB-INF/upload";
330
331     // 临时文件存储目录
332     private static final String TEMP_DIRECTORY = "WEB-INF/temp";
333
334     @Override
335     protected void doPost(HttpServletRequest request, HttpServletResponse
336 response) throws ServletException, IOException {
337     // 检查请求是否为多部分内容
338     if (!ServletFileUpload.isMultipartContent(request)) {
339         // 如果不是, 设置错误信息并转发到 info.jsp
340         request.setAttribute("message", "错误: 表单必须包含 enctype=multipart/form-data");
341         request.getRequestDispatcher("/info.jsp").forward(request, response);
342         return;
343     }
344
345     // 配置上传参数
346     DiskFileItemFactory factory = getDiskFileItemFactory(request);
347     ServletFileUpload upload = getServletFileUpload(factory);

```

```

348         // 设置整体请求的大小限制
349         upload.setSizeMax(MAX_FILE_SIZE);
350
351         // 确保上传目录存在
352         String uploadPath = getServletContext().getRealPath("/") + File.separator + UPLOAD_DIRECTORY;
353         File uploadDir = new File(uploadPath);
354         if (!uploadDir.exists()) {
355             uploadDir.mkdirs();
356         }
357
358         String message = "";
359
360         try {
361             // 解析请求的内容以提取文件数据
362             List<FormItem> formItems = upload.parseRequest(request);
363
364             if (formItems != null && !formItems.isEmpty()) {
365                 // 遍历表单字段
366                 for (FormItem item : formItems) {
367                     // 处理非表单字段
368                     if (item.isFormField()) {
369                         String fieldName = item.getFieldName();
370                         String fieldValue = item.getString("UTF-8");
371                         System.out.println("普通表单字段: " + fieldName +
372 " = " + fieldValue);
373                     } else {
374                         // 处理文件字段
375                         String fileName = Paths.get(item.getName()).getFileName().toString();
376                         if (fileName == null || fileName.trim().isEmpty()) {
377                             message = "未选择文件上传";
378                             continue;
379                         }
380
381                         // 检查文件大小
382                         if (item.getSize() > MAX_FILE_SIZE) {
383                             message = "文件大小超过限制 (10MB) ";
384                             continue;
385                         }
386
387                         // 检查文件类型 (根据需要, 可以添加更多类型)
388                         String fileExt = getFileExtension(fileName);
389                         if (!isAllowedFileType(fileExt)) {
390                             message = "不支持的文件类型: " + fileExt;
391                             continue;
392                         }

```



```

392
393 // 生成唯一的文件名, 防止重名
394 String newFileName = UUID.randomUUID().toString()
    + "_" + fileName;
395
396 // 保存文件到指定路径
397 File storeFile = new File(uploadDir, newFileName)
;
398 try (InputStream inputStream = item.getInputStrea
399 m());
    FileOutputStream outputStream = new FileOutp
400 utStream(storeFile)) {
401     byte[] buffer = new byte[1024];
402     int bytesRead;
403     while ((bytesRead = inputStream.read(buffer))
    != -1) {
404         outputStream.write(buffer, 0, bytesRead);
405     }
406 }
407
408 message = "文件上传成功: " + newFileName;
409 System.out.println(message);
410 }
411 }
412 } catch (FileUploadException e) {
413     message = "错误信息: " + e.getMessage();
414     e.printStackTrace();
415 }
416
417 // 将消息存储在请求范围内
418 request.setAttribute("message", message);
419 // 转发到 info.jsp
420 request.getRequestDispatcher("/info.jsp").forward(request, respon
421 se);
422 }
423
424 /**
425  * 配置 DiskFileItemFactory
426  */
427 private DiskFileItemFactory getDiskFileItemFactory(HttpServletRequest
428 request) {
429     // 配置工厂参数
430     DiskFileItemFactory factory = new DiskFileItemFactory();
431     // 设置内存缓冲区大小
432     factory.setSizeThreshold(MEMORY_THRESHOLD);
433     // 设置临时文件存储位置

```

```

443         String tempPath = getServletContext().getRealPath("/") + File.separator + TEMP_DIRECTORY;
444         File tempDir = new File(tempPath);
445         if (!tempDir.exists()) {
446             tempDir.mkdirs();
447         }
448         factory.setRepository(tempDir);
449
450         return factory;
451     }
452
453     /**
454      * 配置 ServletFileUpload
455      */
456     private ServletFileUpload getServletFileUpload(DiskFileItemFactory factory) {
457         return new ServletFileUpload(factory);
458     }
459
460     /**
461      * 获取文件扩展名
462      */
463     private String getFileExtension(String fileName) {
464         if (fileName == null || fileName.trim().isEmpty()) {
465             return "";
466         }
467         int dotIndex = fileName.lastIndexOf(".");
468         return (dotIndex == -1) ? "" : fileName.substring(dotIndex + 1).toLowerCase();
469     }
470
471     /**
472      * 检查是否为允许的文件类型
473      */
474     private boolean isAllowedFileType(String fileExt) {
475         // 定义允许的文件类型（根据需求添加）
476         String[] allowedTypes = {"jpg", "jpeg", "png", "gif", "pdf", "doc", "docx"};
477         for (String type : allowedTypes) {
478             if (type.equalsIgnoreCase(fileExt)) {
479                 return true;
480             }
481         }
482         return false;
483     }
484
485     @Override

```

```
477     protected void doGet(HttpServletRequest request, HttpServletResponse
478 response) throws ServletException, IOException {
479         // 直接跳转到上传页面（根据实际情况调整）
480         response.sendRedirect("upload.jsp");
481     }
482 }
```